Lab 02 Template - Ethan Roepke

Part 01)

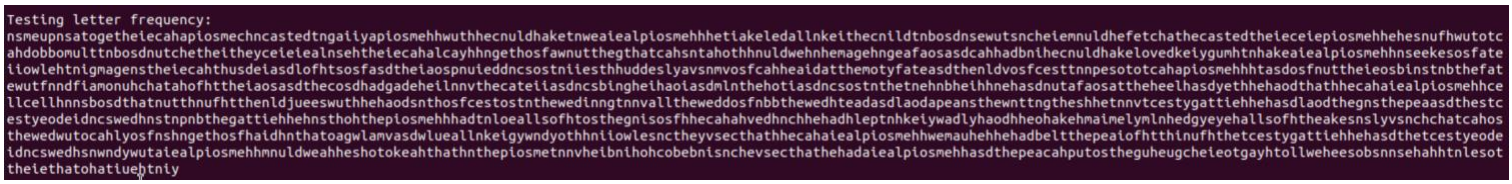
1) **Submit your commented code from Part one as "Lab02_part01.py" to canvas. It needs to be documented and it needs to run correctly.**
   (20 points)
   SUBMITTED
2) **What is the plaintext message?**
   (10 points)
   onceuponatimetherewasaprincewhowantedtomarryaprincessbutshewouldhavetobearealprincesshetraveledallovertheworldtofindonebutnowherecouldhegetwhathewantedtherewereprincessesenoughbutitwasdifficulttofindoutwhethertheywererealonestherewasalwayssomethingaboutthemthatwasnotasitshouldbesohecamehomeagainandwassadforhewouldhavelikedverymuchtohavearealprincessoneeveningaterriblestormcameontherewasthunderandlightningandtherainpoureddownintorrentssuddenlyaknockingwasheardatthecitygateandtheoldkingwenttoopenitwasaprincessstandingoutthereinfrontofthegatebutgoodgraciouswhatasighttherainandthewindhadmadeherlookthewaterrandownfromherhairandclothesitrandownintothetoesofhershoesandoutagainattheheelsandyetshesaidthatshewasarealprincesswellwellsoonfindthatoutthoughttheoldjueenbutshesaidnothingwentintothebedroomtookallthebeddingoffthebedsteadandlaidapeaonthebottomthenshetooktwentymattressesandlaidthemonthepeaandthentwentyeiderdownbedsontopofthemattressesonthistheprincesshadtoliеallnightinthemorningshewasaskedhowshehadsleptohverybadlysaidsheihavescarcelyclosedmyeyesallnightheavenonlyknowswhatwasinthebedbutiwaslyingonsomethinghardsothatiamblackandblueallovermybodyitshorriblenowtheyknewthatshewasarealprincessbecauseshehadfeltthepearightthroughthetwentymattressesandthetwentyeiderdownbedsnobodybutarealprincesscouldbeassensitiveasthatsotheprincetookherforhiswifefornowheknewthathehadarealprincessandthepeawasputinthemuseumwhereitmaystillbeseenifnoonehasstolenittherethatisatruestory

3) **Documentation of the iterations to get to the plaintext message.**
   (15 points)

```
Testing letter frequency:
nimeupniathgetoerecasaprhimeconcaitedtngarryaprhimesswutsoecnuldoaketnwearealprhimessoetrakeledallnkertoecnrldtnbhidniewutincoeremnuldoefetcoatoecaitedtoerecereprhimesseseinufowuthtcasdhbbhmulttnbhidnutcoetoertoeycererealniestoerecasalcayssngetohifawnuttoegtoatcasintashtsonuldwesnoemageongeafahiaidcassadbnroecnuldoakelhvedkerygumotnoakearealprhimessnieekeihifaterrhwlestnrgmagenitoerecastouideraidlhfotihifaidtoerahipnureddncihitnrreitssuddeilyavinmvhifcasoeardattoemhtyfateaidtoenldvhifceittnnpeihthtcasaprhimessstaidhifnuttoerehibrnitnbtoefatewutfnndframhnuscoatashfottoerahiaidtoechidoadgadeoerlnnvtoecaterraidncibrngoeroahraidmlntoeshtraidncihitntoetnesnboersonesaidnutafahiattoeoeelsaidyetsoesahdtoatsoecasarealprhimesscellcellsnnibhidtoatnuttonufottoenldjueeiwutsoesahdintohifceithitntoewedrnngtnnvalltoeweddhifnbbtoewedsteadaidlahdapeanitoewnttngtoeisoetnnvtceitygattressesaidlahdtoegnitoepeaaidtoeitceityehderdnciwedsnitnpnbtoegattressesnitohstoeprhimessoadtnlheallihfothitoegnrihifsoecasasvedoncsoeoadsleptnokerywadlysahdsoehoakesmarmelymlnsedgyeyesallihfotoeakeinilyvincscoatcashltoewedwuthcaslyhifnisngetohifoardsntoathagwlamvaidwlueallnkergywndyhtsonrrhwleinctoeyviectoatsoecasarealprhimesswemausesoeoadbelttoepearhfottornufotoetceitygattressesaidtoetceityehderdnciwedsinwndywutarealprhimessmnuldweasseishthkeastoatsntoeprhimetnnvoerbnrohschbebnrincoeviectoatoeoadarealprhimessaidtoepeacasputhitoeguseugcoerehtgaysthllweseeihbinnieoasstnleihttoeretoathsatruestnry
```

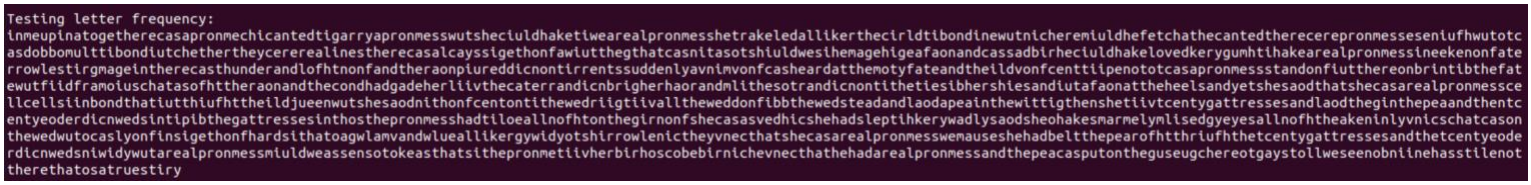
decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "o").replace("J", "i").replace("K", "n").replace("X", "s").replace("S", "h").replace("W", "r").replace("U", "d").replace("G", "l").replace("N", "c").replace("V", "u").replace("Y", "m").replace("T", "w").replace("P", "f").replace("H", "g").replace("Z", "y").replace("L", "p").replace("O", "b").replace("F", "v").replace("B", "k").replace("M", "j")

This is my first run, I see "to" and "toe" show up a bunch



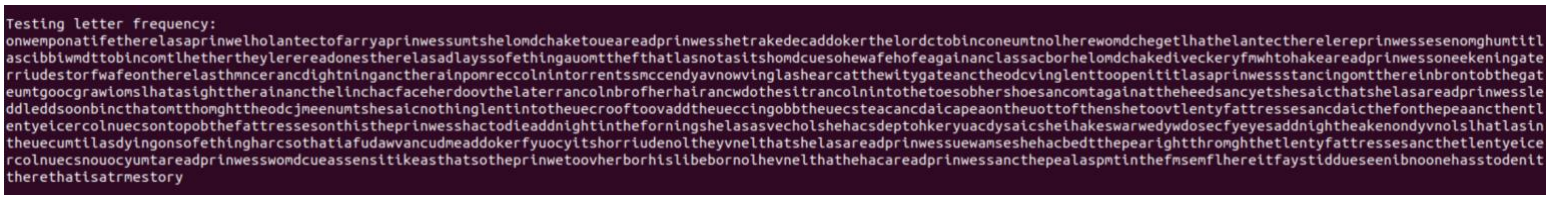
decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "s").replace("K", "n").replace("X", "h").replace("S", "o").replace("W", "i").replace("U", "d").replace("G", "l").replace("N", "c").replace("V", "u").replace("Y", "m").replace("T", "w").replace("P", "f").replace("H", "g").replace("Z", "y").replace("L", "p").replace("O", "b").replace("F", "v").replace("B", "k").replace("M", "j")

This is my second run, I believe I found the word “the”, also found words like “that”, “to” Confident that “t, h, e, a, o” are correct



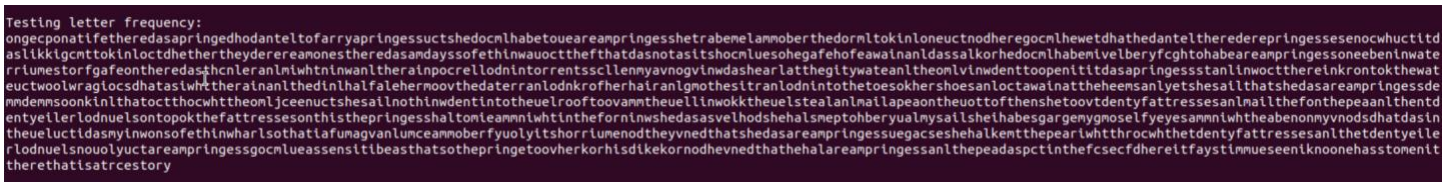
decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "n").replace("K", "i").replace("X", "s").replace("S", "o").replace("W", "r").replace("U", "d").replace("G", "l").replace("N", "c").replace("V", "u").replace("Y", "m").replace("T", "w").replace("P", "f").replace("H", "g").replace("Z", "y").replace("L", "p").replace("O", "b").replace("F", "v").replace("B", "k").replace("M", "j")

This is my third run, I am starting to see sentences starting to get together. For instance the first sentence is coming together with “in me up ina together” also “there, they”. I am confident in letters “t, h, e, a, o, n” y possibly correct.



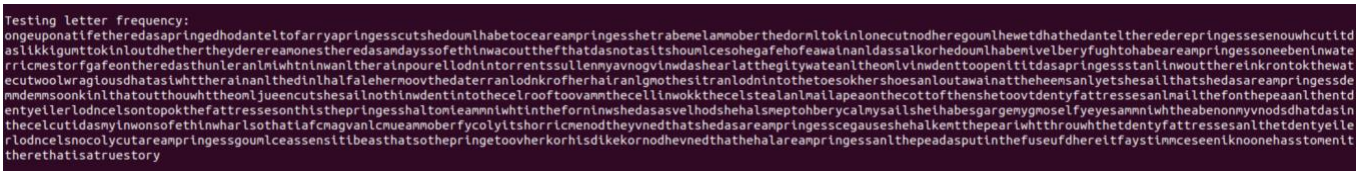
decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "n").replace("K", "o").replace("X", "s").replace("S", "i").replace("W", "r").replace("U", "c").replace("G", "d").replace("N", "l").replace("V", "m").replace("Y", "w").replace("T", "u").replace("P", "g").replace("H", "f").replace("Z", "y").replace("L", "p").replace("O", "b").replace("F", "v").replace("B", "k").replace("M", "j")

This is my fourth run, I am getting more words that I believe are right, this includes “story” “soon”. I am confident on the letters “s”. I believe I have the first 9 characters mapped right.



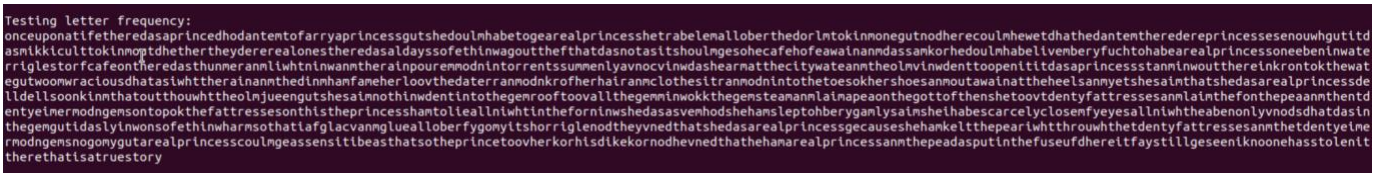
decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "n").replace("K", "o").replace("X", "s").replace("S", "i").replace("W", "r").replace("U", "l").replace("G", "m").replace("N", "d").replace("V", "c").replace("Y", "g").replace("T", "u").replace("P", "w").replace("H", "f").replace("Z", "y").replace("L", "p").replace("O", "k").replace("F", "v").replace("B", "b").replace("M", "j")

This is my fifth run, I am starting to see words coming together and can start guessing more words more confident as they become more obvious. For example, the ending right now says “thatisatrcestory”. I can see that this will translate to “thatisatruestory”. So next step ill change “c” to “u” and hopefully I get more words to show.



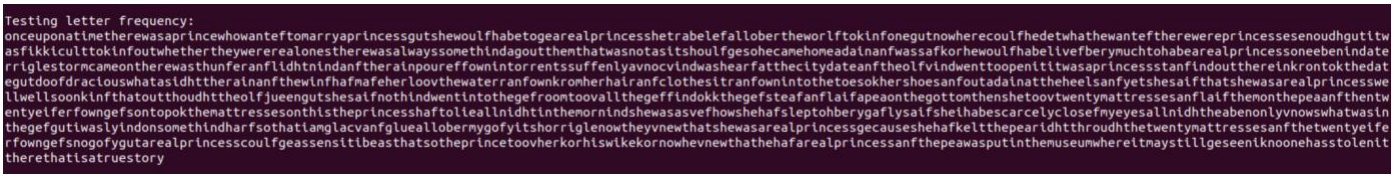
decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "n").replace("K", "o").replace("X", "s").replace("S", "i").replace("W", "r").replace("U", "l").replace("G", "m").replace("N", "d").replace("V", "u").replace("Y", "g").replace("T", "c").replace("P", "w").replace("H", "f").replace("Z", "y").replace("L", "p").replace("O", "k").replace("F", "v").replace("B", "b").replace("M", "j")

This is run number six, I was right on “V” mapping to “u”. I cracked words “true” “upon” possibly “out”



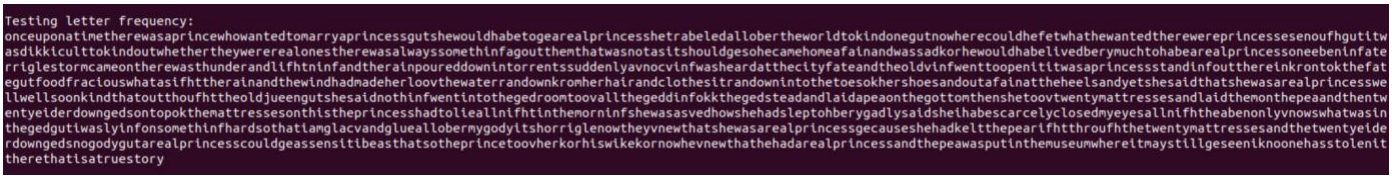
decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "n").replace("K", "o").replace("X", "s").replace("S", "i").replace("W", "r").replace("U", "m").replace("G", "l").replace("N", "d").replace("V", "u").replace("Y", "c").replace("T", "g").replace("P", "w").replace("H", "f").replace("Z", "y").replace("L", "p").replace("O", "k").replace("F", "v").replace("B", "b").replace("M", "j")

Run number seven, looks like I have “onceuponatifetheredasaprince” this helps me to correct the missing letters in this piece of sentence. Theoretically, this should say “onceuponatimetherewasaprince” I will change “f” to “m” and “d” to “w”



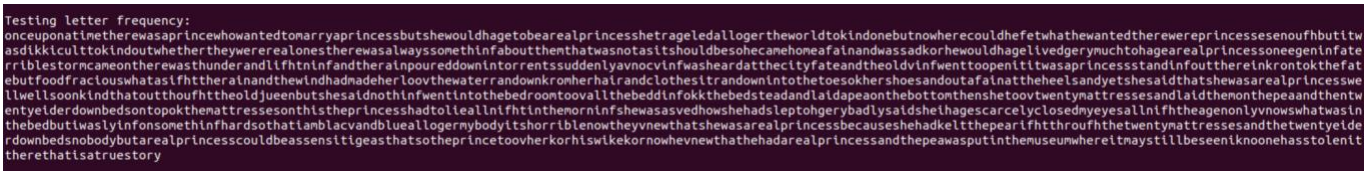
decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "n").replace("K", "o").replace("X", "s").replace("S", "i").replace("W", "r").replace("U", "f").replace("G", "l").replace("N", "w").replace("V", "u").replace("Y", "c").replace("T", "g").replace("P", "d").replace("H", "m").replace("Z", "y").replace("L", "p").replace("O", "k").replace("F", "v").replace("B", "b").replace("M", "j")

This is my run number eight, I was correct on the letter swaps from previous run. Now I can assume we flip “f” to “d”, as I can see this will make words “wanted” “would” “could”



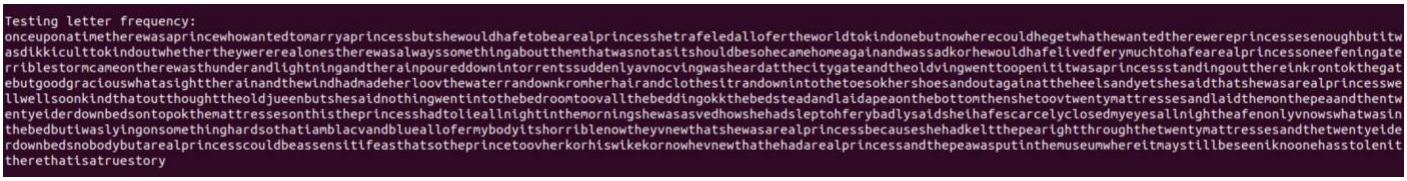
decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "n").replace("K", "o").replace("X", "s").replace("S", "i").replace("W", "r").replace("U", "d").replace("G", "l").replace("N", "w").replace("V", "u").replace("Y", "c").replace("T", "g").replace("P", "f").replace("H", "m").replace("Z", "y").replace("L", "p").replace("O", "k").replace("F", "v").replace("B", "b").replace("M", "j")

Run number nine, I was correct “U” maps to “d”. Going down the sentence “marryaprincessgutshewouldhabeto”. With common guess I would change “g” to “b” to get the word “but”



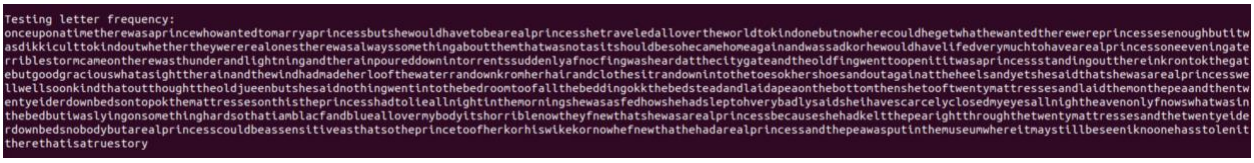
decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "n").replace("K", "o").replace("X", "s").replace("S", "i").replace("W", "r").replace("U", "d").replace("G", "l").replace("N", "w").replace("V", "u").replace("Y", "c").replace("T", "b").replace("P", "f").replace("H", "m").replace("Z", "y").replace("L", "p").replace("O", "k").replace("F", "v").replace("B", "g").replace("M", "j")

Run ten, we are so close to decrypting the cipher text. A section that pops out is “thebedbutiwaslyinfonsomethinf”. This shows me to change “f” to “g”. the sentence would be “thebedbutiwaslyingonsomething”



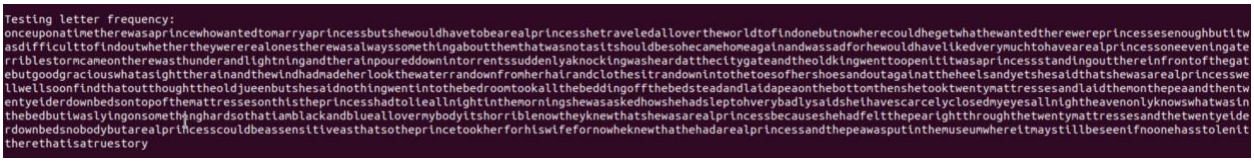
decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "n").replace("K", "o").replace("X", "s").replace("S", "i").replace ("W", "r").replace("U", "d").replace("G", "l").replace("N", "w").replace("V", "u").replace("Y", "c").replace("T", "b").replace("P", "g").replace("H", "m").replace("Z", "y").replace ("L", "p").replace("O", "k").replace("F", "v").replace("B", "f").replace("M", "j")

Run eleven, I should only have a few more test to finish up. I need to change “f” to “v”



decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "n").replace("K", "o").replace("X", "s").replace("S", "i").replace ("W", "r").replace("U", "d").replace("G", "l").replace("N", "w").replace("V", "u").replace("Y", "c").replace("T", "b").replace("P", "g").replace("H", "m").replace("Z", "y").replace ("L", "p").replace("O", "k").replace("F", "f").replace("B", "v").replace("M", "j")

Run twelve, I will flip “k” to “f” to make words “find”



decryptTest = content.replace("I", "e").replace("C", "t").replace("R", "a").replace("Q", "h").replace("J", "n").replace("K", "o").replace("X", "s").replace("S", "i").replace ("W", "r").replace("U", "d").replace("G", "l").replace("N", "w").replace("V", "u").replace("Y", "c").replace("T", "b").replace("P", "g").replace("H", "m").replace("Z", "y").replace ("L", "p").replace("O", "f").replace("F", "k").replace("B", "v").replace("M", "j")

After 13 tries I have cracked the plaintext password!!

4) **Beside the English language frequency of each character what else could you have calculated to help you find the plaintext?**
(10 points)

Besides using the language frequency of each character. We could have used a few other options, the main option to help solve the plaintext is the Index of Coincidence. In class we learned that with a special equation, it will measure how likely you get two matching letters in a random selection two characters from the text.

$$\left(\left(\frac{n_a}{N} * \frac{n_a - 1}{N - 1}\right) + \left(\frac{n_b}{N} * \frac{n_b - 1}{N - 1}\right) + \cdots \left(\frac{n_z}{N} * \frac{n_z - 1}{N - 1}\right)\right)$$

I used this personally for solving the plaintext in lab and relates to language frequency but we can use bigrams and trigrams to solve the plaintext as well.

a. Explain the difference between the sliding window method and the block method. (10 points)
Sliding window method will continue to process data and move over an array/list of characters/bits one at a time. While sliding, a new step involves processing a new portion of the input, but still overlapping with the previous input.
Block method is predefined size and processes all at once. So when decrypting/encrypting, the text will be processed at once unlike the sliding window method that goes one by one.

5) **Explain the difference between conducting an exhaustive key search vs. English language character frequency and explain how your results supported or disproved the number of attempts needed to correctly decrypt the message.**
(10 points)

Conducting an exhaustive key search is to try every possible key until you find the correct one to crack the cipher text. English language character frequency is mapping the most common letter in the alphabet with the most common character in the ciphertext. With my character frequency I got lucky on step 6/7 that gave me many words coming together. After that step I was able to make educated guesses to combine words and came together quickly. This only took me 13 steps which was pretty decent I think, trying to do an exhaustive key search I believe would be more challenging as if it's a much larger key size it can take many more steps than a English frequency. However if the key is very short and you know that then it would be very easy to use the exhaustive key search.

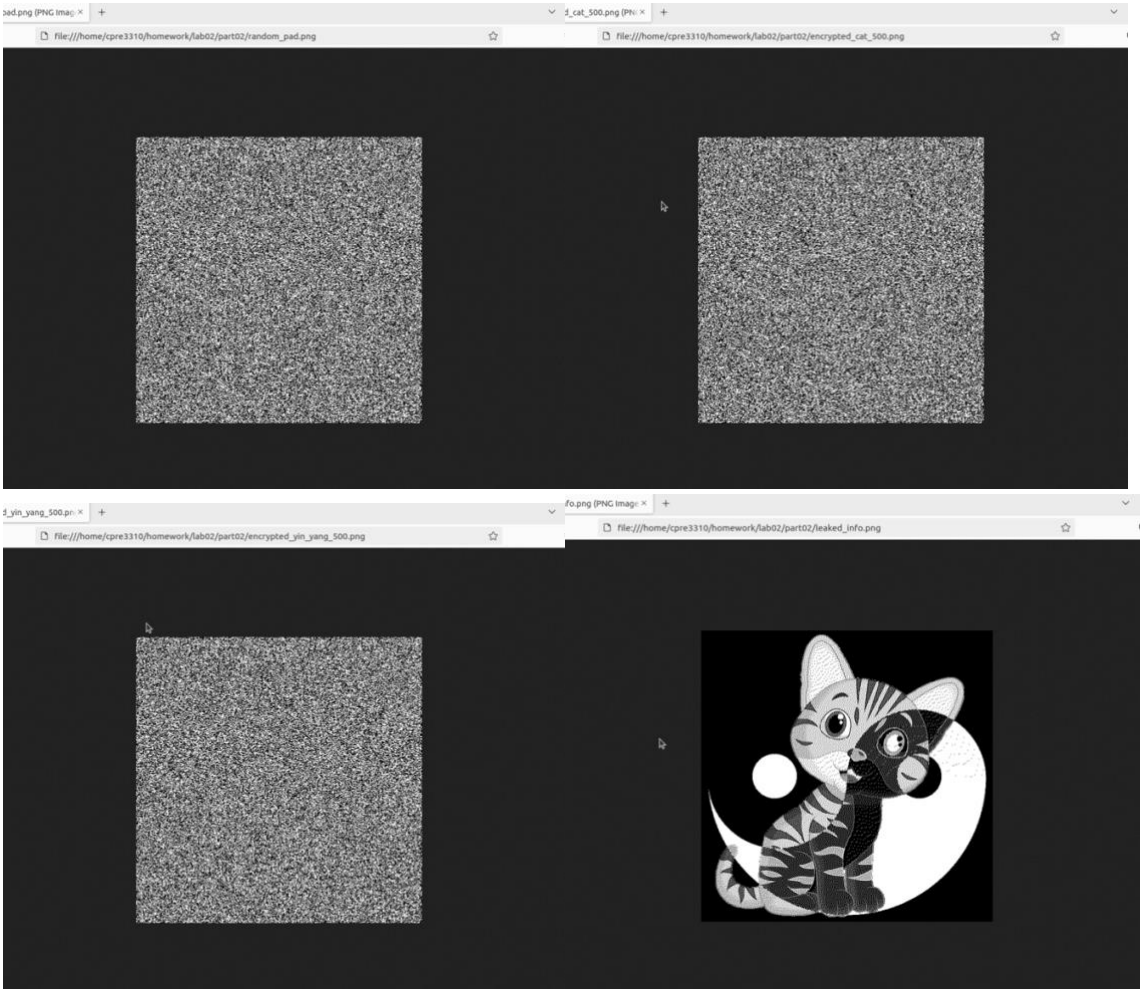
6) **Find a character frequency distribution for another language. Provide the frequency distribution in your lab report *not a link, but the actual distribution) and be sure to identify which language it is.**
(5 points)

Hungarian Character Frequency

Letter	Frequency
E	10.53 %
A	8.98 %
T	7.97 %
L	6.79 %
S	6.20 %
N	5.36 %
K	5.05 %
Z	4.41 %
R	4.27 %
I	3.88 %
O	3.85 %
Á	3.67 %
É	3.52 %
G	3.13 %
M	2.92 %
B	2.15 %
Y	2.12 %
V	2.03 %
D	1.73 %
H	1.49 %
J	1.12 %
Ó	1.08 %
Ö	1.08 %
F	1.06 %
P	1.04 %
Ő	1.01 %
U	0.93 %
C	0.81 %
Í	0.64 %
Ü	0.53 %
Ú	0.29 %
Ű	0.23 %
X	0.07 %
W	0.04 %
Q	< 0.01 %

Part 02

7) **Include screenshots of random_pad.png, encrypted_cat_500.pn, encrypted_yin_yang_500.png, and leaked_info.png in your report.**
(10 points)



8) **What do you observe in leaked_info.png? Why does this happen?**
(10 points)

While observing the leaked_infor.png, we see that the image shows the yin yang image and cat image combined into one image. The XOR operation when using two encrypted images combines the pixels data. Both of these images will combine the pixel differences and lead to an overlay of both images creating both images to be visible. This also does decrypt both images so we would be able to see both encrypted images separately.